On the Pathological Behavior of Adaptive Differential Evolution on Hybrid Objective Functions

Ryoji Tanabe
Graduate School of Arts and Sciences
The University of Tokyo
Tokyo, Japan
rt.ryoji.tanabe@gmail.com

Alex S. Fukunaga
Graduate School of Arts and Sciences
The University of Tokyo
Tokyo, Japan
fukunaga@idea.c.u-tokyo.ac.jp

ABSTRACT

Most state-of-the-art Differential Evolution (DE) algorithms are adaptive DEs with online parameter adaptation. We investigate the behavior of adaptive DE on a class of hybrid functions, where independent groups of variables are associated with different component objective functions. An experimental evaluation of 3 state-of-the-art adaptive DEs (JADE, SHADE, jDE) shows that hybrid functions are "adaptive-DE-hard". That is, adaptive DEs have significant failure rates on these new functions. In-depth analysis of the adaptive behavior of the DEs reveals that their parameter adaptation mechanisms behave in a pathological manner on this class of problems, resulting in over-adaptation for one of the components of the hybrids and poor overall performance. Thus, this class of deceptive benchmarks pose a significant challenge for DE.

Categories and Subject Descriptors

G.1.6 [**Mathematics of Computing**]: Optimization—*Global optimization*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

General Terms

Algorithms

Keywords

Adaptive Differential Evolution, Parameter Control, Deception, Hybrid Functions

1. INTRODUCTION

Differential Evolution (DE) is an Evolutionary Algorithm (EA) that was primarily designed for real parameter optimization problems [11]. Despite its relative simplicity, DE has been shown to be competitive with more complex optimization algorithms, and has been applied to many practical problems [7]. As with other EAs, the search performance of DE algorithms depends on control parameter settings [6, 7, 10]. A standard DE has three main control parameters, which are the population size N, scaling factor F, and crossover rate CR. However, it is well-known that the optimal settings of these parameters are problem-dependent and parameter tuning in real-world problem is often infeasible for various reasons. Since this is a significant problem in practice, adaptive mechanisms for adjusting the DE control parameters on-line during the search process have been studied by many researchers [6, 13, 18].

It is well known that when the objective function is unimodal and nonseparable, high values of CR tend to perform well, while low values of CR tend to be effective for problems that are multimodal and separable.[1] In addition, when the objective is unimodal, moderate values of F work well, but if the objective is multimodal, high F is desirable in order to maintain diversity. Thus, adaptive DE algorithms attempt to adapt control parameters in order to match the characteristics of the objective function [6, 13, 18].

EA performance evaluations for black-box optimization are usually based on a suite of artificial benchmark problems. In recent years, benchmark sets such as the GECCO BBOB [1] and CEC2005 benchmarks [12] have been widely used in the EA community. However, it has been noted that these benchmark sets consist mostly of problems which have unrealistic, "extreme" characteristics, e.g., all of the variables are separable, or all of the variables are nonseparable [14]. Many real-world problems have more complex structure and can not be trivially characterized. For example, problems can be partially separable, where some groups of variables interact with each other but are independent of the other groups, i.e., "separability" is not a binary feature. Yet, previous standard benchmarks focused almost exclusively on such extreme benchmarks, casting some doubt on the utility of such benchmarks in predicting the performance of EAs on real-world problems. For example, real-world problems where all variables interact uniformly and completely with each other (as in the standard Rosenbrock and Rotated-Rastrigin functions) are rare. Because of this growing realization, the EA community has recently introduced benchmark sets that incorporate more complex interactions between variables [9, 14].

Hybrid objective functions [9] are a class of benchmark

GECCO'14, July 12-16, 2014, Vancouver, BC, Canada.

[1] In *separable* functions the solution vector can be decomposed as $f(\boldsymbol{x}) = \sum_{i=1}^{D} f_i(x_i)$ and can be each dimension can be optimized independently. On the other hand, nonseparable functions have dependences between variables and cannot be so easily decomposed and optimized.

functions that have been recently proposed and incorporated into the CEC2014 single-objective optimization competition test suite. In a hybrid function, the variables in a solution vector are partitioned into multiple groups, and each group is evaluated according to a different component evaluation function. The overall objective function value is the sum of the component objective values. Unlike previous benchmark functions where all variables were associated with the same search space, each group of variables are associated with a search space with characteristics that are distinct from the search spaces associated with the other variable groups.

In this paper, we show that state-of-the-art adaptive DE algorithms such as SHADE [13], JADE [18], and jDE [6] perform extremely poorly on a class of hybrid functions [9] composed of 2 components. We show, for example that a simple hybrid function which, in some sense, closely resembles the well know, trivial Sphere function, poses a serious challenge for adaptive DEs. An in-depth analysis of the failure of DEs on hybrid functions, and show extensive evidence that adaptive DEs are easily deceived by hybrid functions – they tend to quickly adapt to solve the component of the hybrid function that is easier to make progress on, and are unable to solve the other component. Thus, this study identifies a class of *adaptive-DE-hard* problems that confound adaptive DEs and poses a challenging direction for future work.

2. DIFFERENTIAL EVOLUTION

This section briefly describes DE [11]. A DE population is represented as a set of real parameter vectors $\boldsymbol{x}_i = (x_1, ..., x_D)$, $i = 1, ..., N$, where D is the dimensionality of the target problem, and N is the population size.

In each generation t, a mutant vector $\boldsymbol{v}_{i,t}$ is generated from an existing population member $\boldsymbol{x}_{i,t}$ by applying some mutation strategy. $\boldsymbol{v}_{i,t} = \boldsymbol{x}_{r_1,t} + F \cdot (\boldsymbol{x}_{r_2,t} - \boldsymbol{x}_{r_3,t})$ is rand/1 mutation strategy which is the most popular mutation strategy. The indices r_1, r_2, r_3 are randomly selected from $[1, N]$ such that they differ from each other as well as i. The parameter $F \in [0, 1]$ controls the magnitude of the differential mutation operator. After generating the mutant vector $\boldsymbol{v}_{i,t}$, it is crossed with the parent $\boldsymbol{x}_{i,t}$ in order to generate trial vector $\boldsymbol{u}_{i,t}$. Binomial Crossover, the most commonly used crossover operator in DE, is implemented as follows: For each j ($j = 1, ..., D$), if rand$[0, 1) \leq CR$ or $j = j_{rand}$, $u_{j,i,t} = v_{j,i,t}$. Otherwise, $u_{j,i,t} = x_{j,i,t}$. Where, rand$[0, 1)$ denotes a uniformly selected random number from $[0, 1)$, and j_{rand} is a decision variable index which is uniformly randomly selected from $[1, D]$. $CR \in [0, 1]$ is the crossover rate.

After all of the trial vectors $\boldsymbol{u}_{i,t}, 0 \leq i \leq N$ have been generated, each individual $\boldsymbol{x}_{i,t}$ is compared with its corresponding trial vector $\boldsymbol{u}_{i,t}$, keeping the better vector in the population. In Section 3, we say that a generation of trial vector is successful if this replacement occurs. These process are repeated until some termination criterion is encountered.

3. STATE-OF-THE-ART ADAPTIVE DE ALGORITHMS

This section reviews three state-of-the-art adaptive DE variants, jDE [6], JADE [18], SHADE [13] that we evaluate in our experimental study. These three algorithms were selected because jDE and JADE are among the most highly cited adaptive DE algorithms, and SHADE is the hightest ranking DE method in the CEC2013 Competition on Real-Parameter Single Objective Optimization [4].

jDE [6] assigns a different set of parameter values F_i and CR_i to each $\boldsymbol{x}_i$, which is used for generating the trial vectors. Initially, the parameters for all individuals $\boldsymbol{x}_i$ are set to $F_i = 0.5$, $CR_i = 0.9$. The control parameter values of the trial vectors are inherited from their parents. However, each parameter is randomly modified (within a pre-specified range) with some probability, and modified parameters are kept for the next generation only when a trial is successful.

JADE [18] has two corresponding, adaptive variables, μ_{CR}, μ_F. The CR and F associated with each individual are generated according to a normal/Cauchy distribution with means μ_{CR}, μ_F. At the end of each generation, the values of μ_{CR}, μ_F are updated according to the CR, F pair that resulted in the generation of the successful trial vector in that generation. As the search progresses, μ_{CR}, μ_F should gradually approach the optimal values for the given problem. In addition to parameter adaptation, JADE also uses a novel mutation strategy called current-to-$p$best/1 and an external archive for storing previously generated individuals.

SHADE [13] is an adaptive DE which based on JADE algorithm, but uses a historical memory of successful parameter settings based parameter adaptation scheme. Success-history based adaptation uses a historical memory M_{CR}, M_F which stores a set of CR, F values that have performed well in the past, and generate new CR, F pairs by directly sampling the parameter space close to one of these stored pairs.

4. HYBRID FUNCTIONS

In this paper, we study a simplified, special class of hybrid functions [9] in which the number of groups is 2, i.e., each hybrid function is composed of two objective functions f, g and sum the component objectives to obtain the overall, hybrid objective value.

More specifically: Given a D-dimensional decision variable vector $\boldsymbol{x}$, in our hybrid functions $H_{f,g}$, $\boldsymbol{x}$ is partitioned into $\boldsymbol{x}^f$ ($\boldsymbol{x}^f = x^f_1, ..., x^f_{D_f}$) and $\boldsymbol{x}^g$ ($\boldsymbol{x}^g = x^g_1, ..., x^g_{D_g}$), the group of variables which will be evaluated using f, g respectively. Let $r \in [0, 1]$ be the fraction of variables assigned to component objective f. If r is set to high value, $\boldsymbol{x}^f$ is allocated to the high fraction of variables and vice versa. We ensured the integrality of D_f, D_g by setting $D_f = \text{round}(D \times r)$, $D_g = D - D_f$. The D_f indices corresponding to $\boldsymbol{x}^f$ are randomly selected from $[1, D]$, and the remainder are assigned to $\boldsymbol{x}^g$.

The objective function value of a candidate individual $\boldsymbol{x}$ is $H_{f,g}(\boldsymbol{x}) = f(\boldsymbol{x}^f) + g(\boldsymbol{x}^g)$, where $f(\boldsymbol{x}^f)$ and $g(\boldsymbol{x}^g)$ are the result of evaluating $\boldsymbol{x}^f$ and $\boldsymbol{x}^g$ according to f and g, respectively. For example, if f is the Sphere function and g is the Rastrigin function, then the objective value for a candidate solution vector for the 30-dimensional ($D = 30$) hybrid function $H_{\text{Sphere,Rastrigin}}$ with $r = 0.4$ would be the sum of the results of evaluating a 12-variable Sphere function and a 18-variable Rastrigin function.

5. EXPERIMENTS

Our study uses hybrid objective functions that combine pairs of the following five, commonly used benchmark functions: Sphere, Schwefel 1.2, Rosenbrock, Rastrigin, Ackley. Table 1 shows their properties (unimodal/multimodal, separable/nonseparable). The search spaces were normalized to

Table 1: 5 basic functions for hybrid functions

Functions	Properties
Sphere	Separable, Unimodal
Schwefel 1.2	Nonseparable, Unimodal
Rosenbrock	Nonseparable, Weakly Multimodal
Rastrigin	Separable, Strongly Multimodal
Ackley	Separable, Weakly Multimodal

Table 2: 8 hybrid functions

Type of Hybrid	Hybrid Functions	Base Functions
Heterogeneous	$H_{\text{Sph,Ras}}$	Sphere & Rastrigin
	$H_{\text{Sph,Ack}}$	Sphere & Ackley
	$H_{\text{Sch,Ras}}$	Schwefel 1.2 & Rastrigin
	$H_{\text{Sch,Ack}}$	Schwefel 1.2 & Ackley
	$H_{\text{Ros,Ras}}$	Rosenbrock & Rastrigin
	$H_{\text{Ros,Ack}}$	Rosenbrock & Ackley
Homogeneous	$H_{\text{Sph,Sch}}$	Sphere & Schwefel 1.2
	$H_{\text{Ras,Ack}}$	Rastrigin & Ackley

$[-100, 100]^D$ and the global optima were shifted to a point where each dimension is uniformly selected from $[-80, 80]$. See details of these standard functions and their original feasible regions in [16].

Table 2 shows the 8 hybrid functions that we studied, which combine pairs of the 5 basic functions from Table 1. We write $H_{f,g}$ to denotes the hybrid functions constructed from f and g (e.g., $H_{\text{Sph,Ras}}$ is composed of the Sphere and Rastrigin functions). The first 6 hybrid functions in Table 2, which we refer to as *heterogeneous hybrids*, combine functions whose characteristics differ significantly from each other. For example, $H_{\text{Sph,Ras}}$ combines the unimodal Sphere function and the multimodal Rastrigin function. In contrast, the remaining 2 functions, which we refer to as *homogeneous hybrids*, ($H_{\text{Sph,Sch}}$, $H_{\text{Ras,Ack}}$) combine 2 similar functions – $H_{\text{Sph,Sch}}$ combines two unimodal functions, and $H_{\text{Ras,Ack}}$ combines two multimodal functions. The heterogeneous functions are studied in Section 5.1, and the homogeneous functions are studied in Section 5.2.

We studied problems with $D = 30, 50$ dimensions. The ratio of variables assigned to each component function, r (see Section 4), was varied from 0.0 to 1.0 in increments of 0.1. Each run of the DE continues until either (1) the difference between the best-so-far solution and the optimal solution $\leq$ 1e-8, in which case we treat the run as "success", or (2) the number of objective function calls exceeds $D \times 10,000$, in which case the run is treated as a "failure". On each problem, for each r, each algorithm is executed 50 times. The success rate is the number of "successes" (as defined above) divided by 50.

We evaluated three, state-of-the-art adaptive DE algorithms: SHADE [13], JADE [18] and jDE [6] (see Section 3). In addition, we also evaluate the standard DE [11] for comparison. The source code for JADE was from [2], minimally modified these programs so that it would work with our hybrid benchmark functions. The other DE's (standard DE, SHADE, and jDE) were implemented by ourselves.[2] For each algorithm, we used the control parameter values that were suggested in the cited, original papers.[3] For the standard DE, we used a typical configuration [6, 18] with population size $N = 50$ [4], rand/1/bin generation strategy, $F = 0.5$. We set CR to 0.1 and 0.9 since these values have been shown to work well for separable functions and non-separable functions respectively [10]. Below, we refer to the standard DE using CR of 0.1 and 0.9 as DE-CR0.1 and DE-CR0.9, respectively.

5.1 Results for hybrid functions with dissimilar (heterogeneous) components

Figure 1 shows the success rates of DE algorithms on the hybrid functions, $H_{\text{Sph,Ras}}$, $H_{\text{Sph,Ack}}$, $H_{\text{Sch,Ras}}$, $H_{\text{Sch,Ack}}$, $H_{\text{Ros,Ras}}$, $H_{\text{Ros,Ack}}$ (30, 50 dimensions) for various allocations (r) of variables among the two components.

We highlight several trends that are evident in Figure 1:

- The adaptive DE algorithms (SHADE, JADE, jDE) tend to have very high success rates when $r = 0.0$ and 1.0. That is, they are quite capable of solving the pure benchmark functions from which the hybrids are derived. The only exception is the low success rates for jDE on Schwefel 1.2 and Rosenbrock function for 50 dimensions.

- The success rates for adaptive DE algorithms tend to be high when r is close to 0, deteriorate steeply for some particular range of r (most commonly when $r = 0.5 \sim 0.9$) but recover again at $r = 1.0$, and most of the curves in Figure 1 are shaped like a "V" or "U".

- Any degradations in success rates as a function of r that are present in 30 dimensions are even more pronounced in 50 dimensions. This is expected, since problems with higher dimensionality are more difficult for DE algorithms in general.

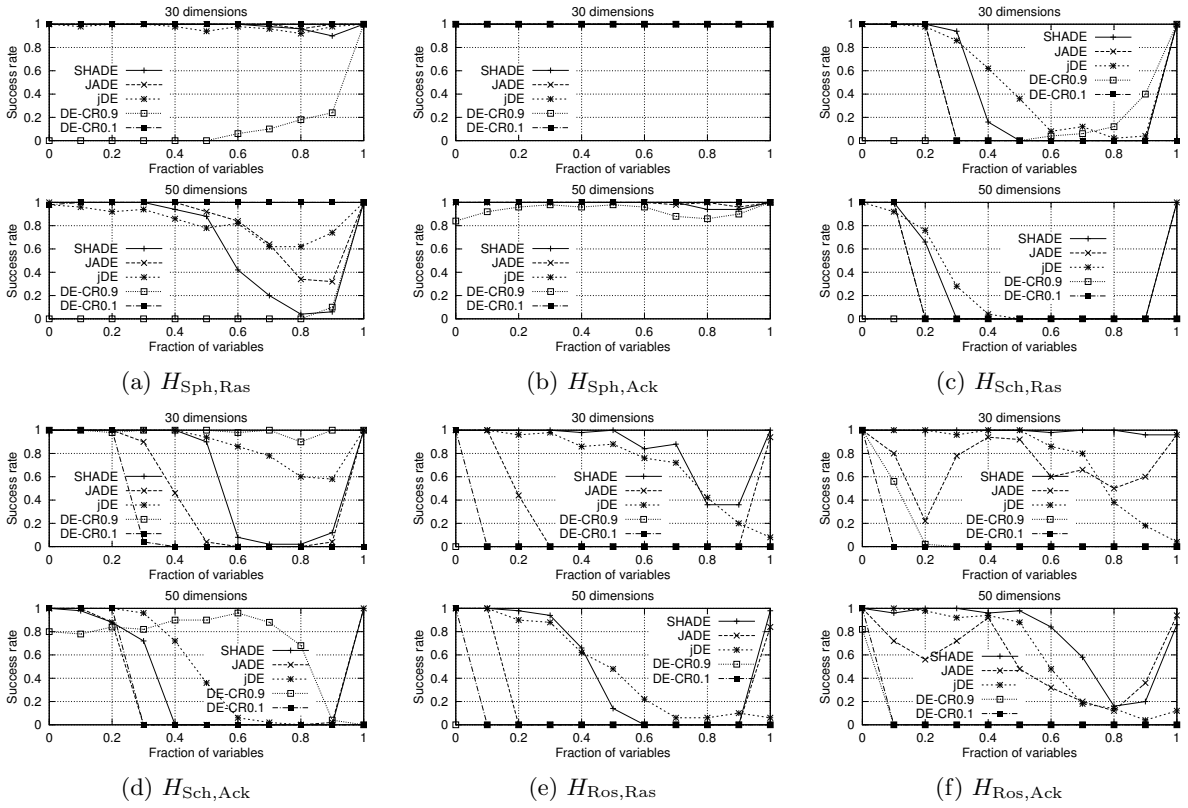
5.1.1 Discussion

It is well known that in general, unimodal functions are easier to search than multimodal functions. Thus, on hybrid functions that combine unimodal and multimodal components ($H_{\text{Sph,Ras}}$, $H_{\text{Sph,Ack}}$, $H_{\text{Sch,Ras}}$, $H_{\text{Sch,Ack}}$), one might expect that as the multimodal component of the hybrid function increases, (i.e., fraction of variables evaluated using the multimodal Rastrigin and Ackley functions increases), the problems would become monotonically more difficult. However, our results indicate the opposite: success rates tended to be higher as the multimodal component increased, and the lowest success rates for SHADE, JADE and jDE occurred when the allocation to unimodal components (Sphere, Schwefel 1.2) were 0.8 and 0.9. A similar tendency was shown in $H_{\text{Ros,Ack}}$, $H_{\text{Ros,Ras}}$, which combines weakly multimodal (Rosenbrock, Ackley) and strongly multimodal (Rastrigin) functions.

[2]We used version 1.1 of SHADE, available at [3].

[3]However, for jDE, we reduced the population size to 50 because using the population size of 100 suggested in [6], since jDE did not converge within the maximum # of fitness evaluations used in this study.

[4]Although $N = 100$ is commonly used, we chose 50 for the same reason as for jDE above.



(a) $H_{\mathrm{Sph,Ras}}$ (b) $H_{\mathrm{Sph,Ack}}$ (c) $H_{\mathrm{Sch,Ras}}$

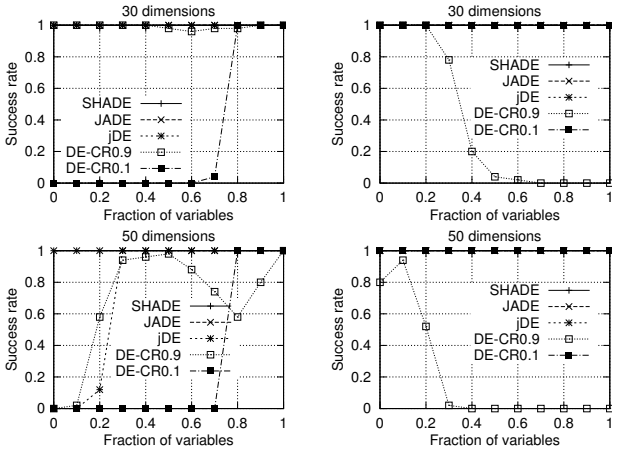
(d) $H_{\mathrm{Sch,Ack}}$ (e) $H_{\mathrm{Ros,Ras}}$ (f) $H_{\mathrm{Ros,Ack}}$

Figure 1: Performance of DE on heterogeneous hybrid functions ($H_{\mathrm{Sph,Ras}}$, $H_{\mathrm{Sph,Ack}}$, $H_{\mathrm{Sch,Ras}}$, $H_{\mathrm{Sch,Ack}}$, $H_{\mathrm{Ros,Ras}}$, $H_{\mathrm{Ros,Ack}}$). Success rate (out of 50 runs) is shown as a function of r, the fraction of variables allocated to component function f. Results for 30 and 50 dimensions are shown.

The results on the $H_{\mathrm{Sph,Ras}}$ hybrid function (Figure 1(a)) are particularly interesting: While the state-of-the-art adaptive DEs that we selected (SHADE, JADE, jDE) have been shown to perform well overall on previous benchmark suites, all of them perform extremely poorly when $r = 0.7 \sim 0.9$ – *In other words, state-of-the-art DE algorithms perform extremely poorly on a hybrid benchmark which is largely composed of the trivial Sphere function!* Furthermore, while DE is known to perform well on separable functions [8], the $H_{\mathrm{Sph,Ras}}$ function is a completely separable function that is extremely challenging for adaptive DE. This is the first such example that we are aware of. Finally (and perhaps most surprisingly), the standard DE using $CR = 0.1$ performed the best among all of the DEs evaluated on $H_{\mathrm{Sph,Ras}}$, significantly outperforming all of the sophisticated, adaptive DEs.

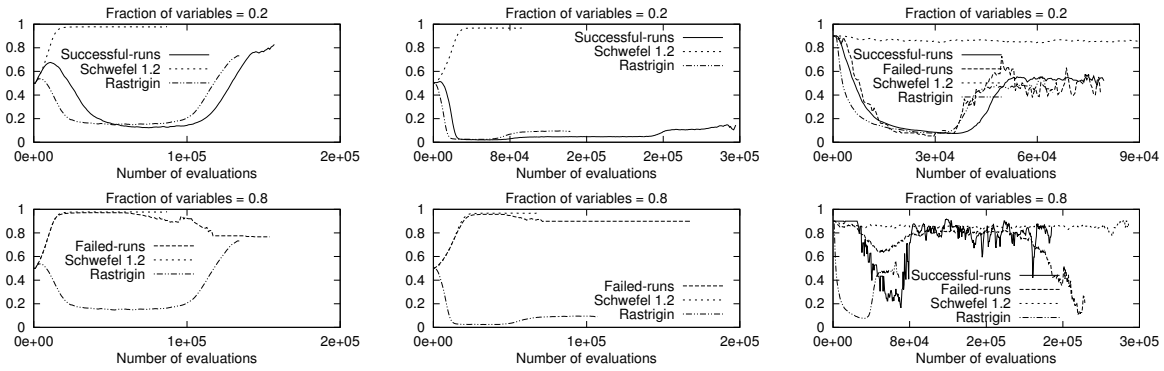
5.2 Results for hybrid functions with similar (homogeneous) components

Section 5.1 showed that on heterogeneous hybrid functions where the components have significantly different search space characteristics (e.g., a unimodal-multimodal hybrid), the performance of DE tended to degrade dramatically for some mixture r of the the components. Are hybrid functions, in general, extremely difficult for DE? In this section, we consider homogeneous hybrid functions where, in contrast to Section 5.1, the component functions have search space



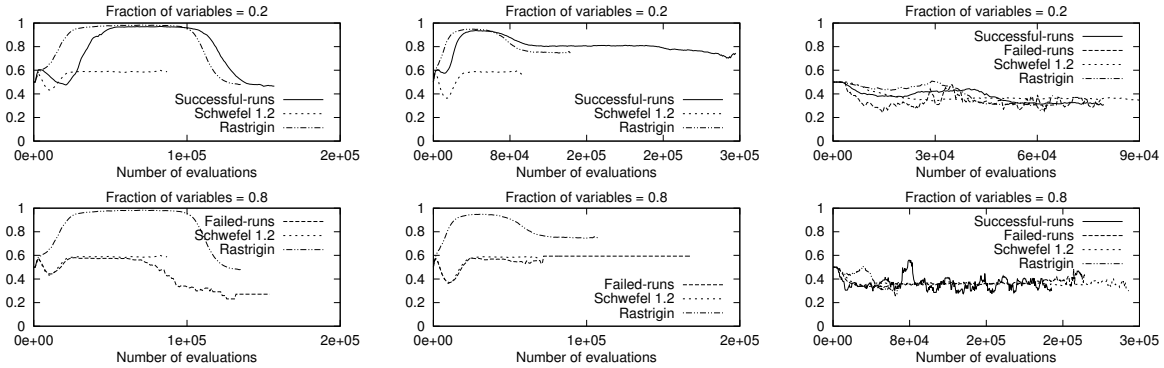
(a) $H_{\mathrm{Sph,Sch}}$ (b) $H_{\mathrm{Ras,Ack}}$

Figure 2: Performance of DE on homogeneous hybrid functions ($H_{\mathrm{Sph,Sch}}$, $H_{\mathrm{Ras,Ack}}$). Success rate (out of 50 runs) is shown as a function of r, the fraction of variables allocated to component function f. Results for 30, 50 dimensions are shown.



(a) SHADE (b) JADE (c) jDE

Figure 3: CR parameter adaptation history of adaptive DEs for $r = 0.2$ (top row) and $r = 0.8$ (bottom row) on $H_{\mathrm{Sch,Ras}}$ ($D = 30$).



(a) SHADE (b) JADE (c) jDE

Figure 4: F parameter adaptation history of adaptive DEs for $r = 0.2$ (top row) and $r = 0.8$ (bottom row) on $H_{\mathrm{Sch,Ras}}$ ($D = 30$).

characteristics that are similar to each other. We used two hybrid functions $H_{\mathrm{Sph,Sch}}$ and $H_{\mathrm{Ras,Ack}}$. $H_{\mathrm{Sph,Sch}}$ combines two unimodal functions, and $H_{\mathrm{Ras,Ack}}$ is a hybrid of two multimodal functions.

The results for $H_{\mathrm{Sph,Sch}}$ and $H_{\mathrm{Ras,Ack}}$ in 30 and 50 dimensions are shown in Figure 2. SHADE, JADE and jDE all achieve almost perfect success rates for all values of r on $H_{\mathrm{Sph,Sch}}$ and $H_{\mathrm{Ras,Ack}}$. This result is somewhat counter-intuitive, given that solving Rastrigin and Ackley (both of which are multimodal) individually is much more difficult than solving Rastrigin and Sphere (unimodal) individually, and yet, the hybrid $H_{\mathrm{Ras,Ack}}$ seems to be much easier than the hybrid function $H_{\mathrm{Sph,Ras}}$ (Figure 1(a)), which stumps all of the adaptive DE algorithms when r is between 0.7-0.9.

This result suggests that while hybrid functions are exceptionally challenging for DE when the component functions have different search space characteristics from each other, hybrids composed of similar components such as $H_{\mathrm{Sph,Sch}}$, $H_{\mathrm{Ras,Ack}}$ do not result in difficult problems for DE.

One notable counterexample to the claim that heterogeneous hybrids are challenging is in Figure 1(b), which shows that all methods perform very well on the hybrid $H_{\mathrm{Sph,Ack}}$, regardless of r, despite the fact that Sphere is unimodal and Ackley is multimodal. However, Ackley is only weakly multimodal, and furthermore, both Sphere and Ackley are separable, so $H_{\mathrm{Sph,Ack}}$ is actually similar to a homogeneous hybrid function such as $H_{\mathrm{Sph,Sch}}$ and $H_{\mathrm{Ras,Ack}}$ and therefore does not pose a challenge for adaptive DE.

5.3 Why do adaptive DEs fail? – Analysis of adaptive DE parameter adaptation on hybrid functions

In order to understand why DE performance degrades so dramatically in the hybrid function experiments in Section 5.1, we analyze the parameter adaptation behavior for the adaptive DEs (SHADE, JADE, jDE) during the runs for the experiments in Section 5.1.[5]

Figures 3 and 4 show the adaptation of the CR, F parameters (see Section 3) by SHADE, JADE, and jDE for $r = 0.2$ (top row) and $r = 0.8$ (bottom row) on $H_{\mathrm{Sch,Ras}}$ (30 dimensions). The x-axis is the number of objective function evaluations. The y-axis shows the proxy values of CR, F

[5]We collected this parameter adaptation data by re-running the experiments described in Section 5.1 with identical random seeds used in the original experiments and saving all of the parameter adaptation histories at every generation.

values for each algorithm. Since each algorithm has a different adaptation mechanism, appropriate proxy values are shown for each mechanism: (1) for SHADE, we show the median values of the CR, F values stored in historical memory, M_{CR}, M_F, [13]; (2) for JADE, μ_{CR}, μ_F [18], the adaptive parameters for CR and F, are shown; (3) for jDE, which assigns a different CR and F to each individual i [6], the median values CR_i and F_i are shown.

Each of 50 runs (for each r, for each function, for each algorithm) was classified as Successful Runs or Failed Runs, depending on whether the run achieved the success criteria defined at the beginning of Section 5 for this study, i.e., error $\leq$ 1e-8. In cases where all 50 runs were failures or all 50 runs were successes, only one of these lines are shown.

Each figure also shows how proxy values for F and CR adapt in successful runs on pure Rastrigin and Schwefel 1.2 functions (i.e., $r = 0.0, 1.0$). The "Schwefel 1.2" and "Rastrigin" lines in Figures 3 and 4 show that on the Schwefel 1.2 function, SHADE, JADE, and jDE adapt CR to high values and F to medium values. On the other hand, for the Rastrigin function, SHADE, JADE, and jDE adapt CR to low values and F converges to high values. Thus, CR and F converge to very different values depending on the target function (Rastrigin vs. Schwefel 1.2). These results are consistent with previous work which experimentally analyzed the dynamics of DE parameter adaptation [6, 10], as well as previous results on parameter adaptation for SHADE, JADE, and jDE [6, 13, 18].

Now, consider the behavior of adaptive DE on $H_{\text{Sch,Ras}}$ when $r = 0.2$ (high fraction of variables allocated to the Rastrigin function). From the figures, the adaptive behavior of CR and F in SHADE and JADE very closely matches their behavior on the Rastrigin function. jDE also displays a similar tendency, albeit weaker (possibly due to the way we defined the proxy metrics as medians). On the other hand, for $r = 0.8$ (high fraction of variables allocated to Schwefel 1.2), the opposite trend can be seen, as the proxies for CR and F behave similarly to the pure Schwefel 1.2 case. This leads to the main, qualitative result of our study:

Observation 1. *When an adaptive DE is applied to a hybrid function, the parameter adaptation mechanism tends to adapt and converge the CR and F parameters so as to match the component for which initial improvements are easier. More specifically, the adaptive mechanisms target the component with larger allocation of variables.*

Observation 1 holds quite robustly for our class of hybrid functions. Trends similar to those seen in Figures 3 and 4 were observed for almost all hybrid functions we tested, for various values of r.

Since limited space prevents us from showing the equivalents of Figures 3 and 4 for all functions and all r, we instead rely on a summary, difference metric which allows to compactly summarize and visualize these trends across all of the hybrid functions, for all r.

Suppose that we run jDE on the Sphere function twice, and that runs #1 and #2 terminate after 5 and 4 generations, respectively, and that the CR-proxy values (as defined above in this section) are (0.7, 0.6, 0.5, 0.7, 0.4) for run #1 and (0.9, 0.8, 0.3, 0.5) for run #2. The average behavior of CR on these two runs is (0.8, 0.7, 0.4, 0.6, 0.4).

Further suppose that jDE is run on Rastrigin twice, with an average CR behavior of (0.4, 0.5, 0.4). The differential values between the CR-history on Sphere and the Rastrigin function is (0.4, 0.2, 0.0), and the distance between the CR-history over time is $(0.4 + 0.2 + 0.0)/3 = 0.2$[6]. Algorithm 1 describes the computation of average distance in general.

Algorithm 1: Function $\delta(\overline{\boldsymbol{p}}_f, \overline{\boldsymbol{p}}_g)$ for computing the difference between two average adaptation histories

input : $\overline{\boldsymbol{p}}_f, \overline{\boldsymbol{p}}_g$: the average adaptation histories on f, g
output: δ: the average distance on overall generations between $\overline{\boldsymbol{p}}_f$ and $\overline{\boldsymbol{p}}_g$

```
t_max = min(|p̄_f|, |p̄_g|), δ = 0;
for t = 1 to t_max do
    δ = δ + |p̄_{f,t} − p̄_{g,t}|;
δ = δ/t_max;
```

Algorithm 2: Aggregating the average distances (to obtain the data points in Figure 5)

input : $\overline{\boldsymbol{p}}_{H_{f,g}}, \overline{\boldsymbol{p}}_f, \overline{\boldsymbol{p}}_g$ on all hybrid functions
output: $\overline{\delta}_{H_{f,g},f}, \overline{\delta}_{H_{f,g},g}$: the average distance on all hybrid functions

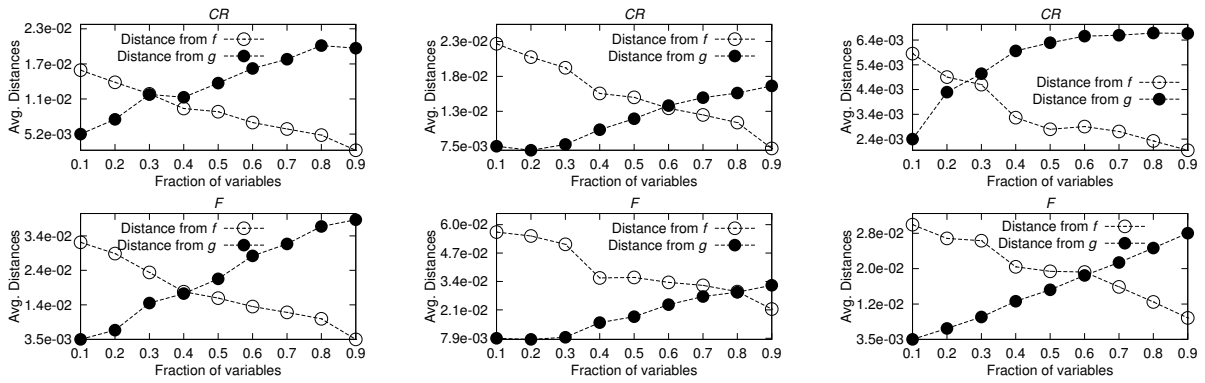
```
δ̄_{H_{f,g},f} = 0, δ̄_{H_{f,g},g} = 0;
for H_{f,g} ∈ I do
    δ̄_{H_{f,g},f} = δ̄_{H_{f,g},f} + δ(p̄_{H_{f,g}}, p̄_f);
    δ̄_{H_{f,g},g} = δ̄_{H_{f,g},g} + δ(p̄_{H_{f,g}}, p̄_g);
δ̄_{H_{f,g},f} = δ̄_{H_{f,g},f}/|I|, δ̄_{H_{f,g},g} = δ̄_{H_{f,g},g}/|I|;
```

The distance $\delta(\overline{\boldsymbol{p}}_f, \overline{\boldsymbol{p}}_g)$ can be used as a measure of the similarity between the parameter adaptation histories $\overline{\boldsymbol{p}}_f, \overline{\boldsymbol{p}}_g$ on f and g. The smaller the value of $\delta(\overline{\boldsymbol{p}}_f, \overline{\boldsymbol{p}}_g)$, the more similar the parameter adaptation histories are, and conversely, the larger $\delta(\overline{\boldsymbol{p}}_f, \overline{\boldsymbol{p}}_g)$ is, the more different the parameter adaptation histories. Average distances among histories among a set of instances $\boldsymbol{I}$ can be computed similarly.

For example, for some particular r, suppose that $\delta(H_{\text{Sph,Ras}}$, Sphere)=0.4 and $\delta(H_{\text{Sch,Ack}}, \text{Schwefel 1.2}) = 0.6$. In this case, the average value of the distances is $(0.4 + 0.6)/2 = 0.5$. For the same r, if $\delta(H_{\text{Sph,Ras}}, \text{Rastrigin}) = 0.6$ and $\delta(H_{\text{Sch,Ack}}, \text{Ackley}) = 0.8$, then the average value of the distances is $(0.6+0.8)/2 = 0.7$. In particular, we are interested in distances between the histories on a hybrid function $H_{f,g}$ with particular r value and the histories on its component functions, $\delta(H_{f,g}, f)$ and $\delta(H_{f,g}, g)$, averaged over all instances $\overline{\delta}^r_{H_{f,g},f}$, $\overline{\delta}^r_{H_{f,g},g}$. Algorithm 2 shows the pseudocode for this overall computation for some particular r.

Figures 5 shows $\overline{\delta}^r_{H_{f,g},f}$, $\overline{\delta}^r_{H_{f,g},g}$ for both DE parameters CR (top row) and F (bottom row), averaged over all 6 of the heterogeneous hybrid functions in Table 2 according to Algorithm 2, for $r \in 0.1, ..., 0.9$. Data for both 30 and 50 dimensions are included in the SHADE and JADE figures (the similarity metric can be computed independently of dimensionality). Since jDE had a success rate of 0 on Schwefel 1.2 for 50-dimensions, we only included data for 30 dimensions. When computing the average histories for the hybrid function, we include all runs, including successful and failed runs. On the other hand, for the average histories for the

[6]If two histories have different lengths, the unmatched portions of the longer history are ignored.



(a) SHADE (b) JADE (c) jDE

Figure 5: Average distances of parameter adaptation history of SHADE, JADE, jDE between heterogeneous hybrid functions ($H_{\mathrm{Sph,Ras}}$, $H_{\mathrm{Sph,Ack}}$, $H_{\mathrm{Sch,Ras}}$, $H_{\mathrm{Sch,Ack}}$, $H_{\mathrm{Ros,Ras}}$, $H_{\mathrm{Ros,Ack}}$) and its component function f, g.

pure component functions f and g, we used only the successful runs, because we want to compare the behavior on hybrid functions vs. the "ideal" behavior on the pure functions, and the average successful behavior is used as an approximation for this "ideal" behavior.

Figures 5 show that for all of the adaptive DEs. When r is low ($H_{f,g}$ is mostly allocated to g), the distance between the behavior on $\overline{\delta}^{r}_{H_{f,g},g}$ is small, which means that the parameter adaptation behavior of the DE on $H_{f,g}$ is quite similar to its parameter adaptation behavior on g. However, as r increases, $\overline{\delta}^{r}_{H_{f,g},g}$ grows monotonically, i.e., the DE behaves differently on the hybrid than it would on g. At the same time, as r increases, $\overline{\delta}^{r}_{H_{f,g},f}$ decreases, and the DE behaves more and more like it would on f. This trend can be very clearly seen for both CR and F.

From the results above, we can see that in a heterogeneous hybrid function $H_{f,g}$ composed of two component functions f, g with different search space characteristics, the proxy values for CR, F in state-of-the-art adaptive DEs behave similarly to when the DE is applied to the individual component function f or g which is allocated more variables in the hybrid function, i.e., the individual component which is most "similar" to the hybrid function.

Thus, we have shown that Observation 1 consistently holds across all 3 state-of-the-art adaptive DEs, for all of the heterogeneous, hybrid functions where pathological behavior was observed in Section 5.1.

5.3.1 Discussion

In general, most adaptive DEs that have been proposed to date, including SHADE, JADE, and jDE, operate according to the principle that if the current set of control parameter setting $S = (CR, F)$ successfully generates trial vectors that are more fit than their parents, then the next set of control parameters to be applied should be somehow based on or influenced by S. In other words, adaptive strategies tend to operate based on a meta-greedy assumption.

On our class of hybrid functions, this meta-greedy approach leads to failure. In a hybrid with 2 component functions, the component C_H with a higher fraction of variables contributes more to the objective function than the other component with lower allocation, C_L, because the overall objective value is the sum of the component objective values. Our results suggest that there is a tendency for adaptive DEs to quickly adapt to achieve good performance on C_H, neglecting C_L. After optimizing the C_H component, the adaptive DE is unable to re-adapt to so that the C_L component can be optimized.

This explains the pathological behaviors by adaptive DEs observed in Section 5.1, Figure 1(c): Although SHADE, JADE, and jDE had success rates close to 1.0 on $H_{\mathrm{Sch,Ras}}$ when $r = 0.2$, the success rate was almost 0.0 when $r = 0.8$. This is most likely due to the meta-greedy nature of the adaptive DE described above. The components of $H_{\mathrm{Sch,Ras}}$, Schwefel 1.2 and Rastrigin, differ significantly. Schwefel 1.2 is unimodal, and nonseparable, while Rastrigin is multimodal and separable, and as noted earlier, parameter settings which are appropriate for one are ineffective for the other. Figures 3 and 4 show that the DEs have adapted entirely for one or the other – this means that they have become poorly adapted for the other component.

When $r = 0.8$, the DEs have adapted to solve Schwefel 1.2, a unimodal, nonseparable function. However, these parameter settings are very poorly suited for Rastrigin, a multimodal, separable function, and it is likely that the DEs fail because their control parameters (which are adapted for a unimodal function) promote rapid descent into local minima, where the search becomes trapped.

A similar conclusion can be reached regarding the failure of adaptive DE on $H_{\mathrm{Ros,Ras}}$ in Figure 1(e). Although both Rosenbrock and Rastrigin are multimodal, Rosenbrock is nonseparable (requiring high CR values), while Rastrigin is separable (requiring low CR values), and overadaptation for one function prevents success on the other component of the hybrid objective. In general, in heterogeneous hybrid functions that include Rastrigin or Ackley, when the allocation of variables to the multimodal Rastrigin and Ackley functions are too small, the adaptive DEs can not sufficiently adapt to the multimodal component, resulting on poor success rates on the hybrid functions.

In contrast, the high success rates for $r = 0.2$ can be explained as follows. Adaptation for the multimodal Rastrigin function does not cripple the DE's ability to solve unimodal problems such as the Schwefel 1.2. Although con-

vergence may be slowed compared to parameter settings that are tuned for a unimodal function, the lack of local minima means that the search algorithm will eventually succeed.

6. CONCLUSION

We investigated the behavior of DE on a class of hybrid functions [9], where groups of variables are assigned to different component functions. We showed that state-of-the-art adaptive DEs (SHADE [13], JADE [18], jDE [6]) all perform very poorly when the components of the hybrid functions are heterogeneous, with different search space characteristics.

The main contribution of this study is an in-depth analysis of adaptive DE control parameter behavior on hybrid functions, showing that adaptive DE fails on these hybrid functions because their parameter adaptation mechanisms are fundamentally mismatched with the structure of hybrid functions. DE control parameter adaptation tends to be "deceived" into over-adapting for one of the components of the hybrid, at the expense of adaptation for the other component.

Previous analysis of parameter adaptation in DE has been limited to plotting control parameter values [6, 18]. Our analysis of DE parameter adaptation introduces a new technique for summarizing and visualizing parameter adaptation (the parameter history distance metric described in Section 5.3) which provided a very clear understanding of when and how DE succeeds and fails in the class of hybrid functions. We believe that this kind of aggregated distance can be applied to understand the adaptive behavior of DE (and other algorithms with adaptive parameters) for other types of functions, e.g. multi-objective optimization.

This work suggests that a parameter adaptation mechanism that seeks to optimize its behavior for a single, target function is a major design limitation when faced with a hybrid objective function. Our preliminary efforts to overcome this limitation by, e.g., associating a different set of (CR, F) parameters for each variable, have not been successful so far, and this seems to be a nontrivial problem. Therefore, designing methods that are effective on hybrid objective functions is a challenging direction for future research. One possible direction would be to investigate crossover methods that explicitly seek to identify and exploit dependencies between variables in order to automatically partition and handle the components of the hybrid objective, e.g., [17].

While this study was limited to hybrid functions with 2 component functions, preliminary experiments suggest that similar issues arise on hybrids with more components, and we are currently extending this study to hybrids with 3+ components. Finally, while this study focused on understanding the difficulty that hybrid functions pose for adaptive DE, heterogeneous hybrid functions are challenging for other kinds of non-adaptive, state-of-the-art DE. CoDE, a well-known non-adaptive DE [15] also fails on these functions (see Supplemental Data [5]). Understanding the reason why hybrid functions are difficult for CoDE, as well as evaluation of hybrid functions on other classes of DE is an avenue for future work.

7. ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grants 2324002 and 25330253.

8. REFERENCES

[1] http://coco.gforge.inria.fr/doku.php.
[2] http://dces.essex.ac.uk/staff/qzhang.
[3] https://sites.google.com/site/tanaberyoji.
[4] http://www.ntu.edu.sg/home/EPNSugan/index_files-/CEC20-13/CEC2013.htm.
[5] Supplementary material of this paper.
[6] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer. Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Tran. Evol. Comput.*, 10(6):646–657, 2006.
[7] S. Das and P. N. Suganthan. Differential Evolution: A Survey of the State-of-the-Art. *IEEE Tran. Evol. Comput.*, 15(1):4–31, 2011.
[8] N. Hansen and S. Kern. Evaluating the CMA Evolution Strategy on Multimodal Test Functions. In *PPSN*, pages 282–291, 2004.
[9] J. J. Liang, B. Y. Qu, and P. N. Suganthan. Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization. Technical report, Zhengzhou University and Nanyang Technological University, 2013.
[10] J. Rönkkönen, S. Kukkonen, and K. V. Price. Real-Parameter optimization with Differential Evolution. In *IEEE CEC*, pages 506–513, 2005.
[11] R. Storn and K. Price. Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J. Global Optimiz.*, 11(4):341–359, 1997.
[12] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari. Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization. Technical report, Nanyang Technological University, 2005.
[13] R. Tanabe and A. Fukunaga. Success-History Based Parameter Adaptation for Differential Evolution. In *IEEE CEC*, pages 71–78, 2013.
[14] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise. Benchmark Functions for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization. Technical report, University of Science and Technology of China, 2010.
[15] Y. Wang, Z. Cai, and Q. Zhang. Differential Evolution With Composite Trial Vector Generation Strategies and Control Parameters. *IEEE Tran. Evol. Comput.*, 15(1):55–66, 2011.
[16] X. Yao, Y. Liu, and G. Lin. Evolutionary Programming Made Faster. *IEEE Tran. Evol. Comput.*, 3(2):82–102, 1999.
[17] M. Zhabitsky and E. Zhabitskaya. Asynchronous Differential Evolution with Adaptive Correlation Matrix. In *GECCO*, pages 455–462, 2013.
[18] J. Zhang and A. C. Sanderson. JADE: Adaptive Differential Evolution With Optional External Archive. *IEEE Tran. Evol. Comput.*, 13(5):945–958, 2009.